

```
entry:
%stktop_4 = alloca i8, i32 60, align 1
%tos = ptrtoint ptr %stktop_4 to i64
%0 = add i64 %tos, 12
%RBP_N.48 = inttoptr i64 %0 to ptr
%1 = add i64 %tos, 20
%RBP_N.40 = inttoptr i64 %1 to ptr
%2 = add i64 %tos, 28
%RBP_N.32 = inttoptr i64 %2 to ptr
%3 = add i64 %tos, 36
%RBP_N.24 = inttoptr i64 %3 to ptr
%4 = add i64 %tos, 44
%RBP_N.16 = inttoptr i64 %4 to ptr
%5 = add i64 %tos, 52
%RBP_N.8 = inttoptr i64 %5 to ptr
%6 = add i64 %tos, 0
%RSP_P.0 = inttoptr i64 %6 to ptr
store i64 3735928559, ptr %RSP_P.0, align 8
%RBP = ptrtoint ptr %RSP_P.0 to i64
store i64 %arg1, ptr %RBP_N.8, align 1
store i64 %arg2, ptr %RBP_N.16, align 1
store i64 %arg3, ptr %RBP_N.24, align 1
store i64 %arg4, ptr %RBP_N.32, align 1
store i64 %arg5, ptr %RBP_N.40, align 1
store i64 %arg6, ptr %RBP_N.48, align 1
store i32 0, ptr %stktop_4, align 1
br label %bb.1
```

```
bb.1:
%7 = load i32, ptr %stktop_4, align 1
%8 = zext i32 %7 to i64
%9 = zext i32 4 to i64
%10 = sub i64 %8, %9
%11 = call { i64, i1 } @llvm.usub.with.overflow.i64(i64 %8, i64 %9)
%CF = extractvalue { i64, i1 } %11, 1
%ZF = icmp eq i64 %10, 0
%highbit = and i64 -9223372036854775808, %10
%SF = icmp ne i64 %highbit, 0
%12 = call { i64, i1 } @llvm.ssub.with.overflow.i64(i64 %8, i64 %9)
%OF = extractvalue { i64, i1 } %12, 1
%13 = and i64 %10, 255
%14 = call i64 @llvm.ctpop.i64(i64 %13)
%15 = and i64 %14, 1
%PF = icmp eq i64 %15, 0
%CmpSFOF_JGE = icmp eq i1 %SF, %OF
br i1 %CmpSFOF_JGE, label %bb.3, label %bb.2
```

T

F

```
bb.2:
%memload = load i64, ptr %RBP_N.8, align 1
%memload1 = load i64, ptr %stktop_4, align 1
%16 = trunc i64 %memload1 to i32
%RCX = sext i32 %16 to i64
%memref-basereg = add i64 %memload, %RCX
%17 = inttoptr i64 %memref-basereg to ptr
%memload2 = load i32, ptr %17, align 1
%18 = trunc i32 %memload2 to i8
%EDX = sext i8 %18 to i32
%memload3 = load i64, ptr %RBP_N.16, align 1
%memload4 = load i64, ptr %stktop_4, align 1
%19 = trunc i64 %memload4 to i32
%RCX5 = sext i32 %19 to i64
%memref-basereg6 = add i64 %memload3, %RCX5
%20 = inttoptr i64 %memref-basereg6 to ptr
%memload7 = load i32, ptr %20, align 1
%21 = trunc i32 %memload7 to i8
%ESI = sext i8 %21 to i32
%EDX11 = add nsw i32 %EDX, %ESI
%highbit8 = and i32 -2147483648, %EDX11
%SF9 = icmp ne i32 %highbit8, 0
%ZF10 = icmp eq i32 %EDX11, 0
%memload12 = load i64, ptr %RBP_N.24, align 1
%memload13 = load i64, ptr %stktop_4, align 1
%22 = trunc i64 %memload13 to i32
%RCX14 = sext i32 %22 to i64
%memref-basereg15 = add i64 %memload12, %RCX14
%23 = trunc i32 %EDX11 to i8
%24 = inttoptr i64 %memref-basereg15 to ptr
store i8 %23, ptr %24, align 1
%memload16 = load i64, ptr %RBP_N.8, align 1
%memload17 = load i64, ptr %stktop_4, align 1
%25 = trunc i64 %memload17 to i32
%RCX18 = sext i32 %25 to i64
%memref-basereg19 = add i64 %memload16, %RCX18
%26 = inttoptr i64 %memref-basereg19 to ptr
%memload20 = load i32, ptr %26, align 1
%27 = trunc i32 %memload20 to i8
%ESI21 = sext i8 %27 to i32
%memload22 = load i64, ptr %RBP_N.16, align 1
%memload23 = load i64, ptr %stktop_4, align 1
%28 = trunc i64 %memload23 to i32
%RCX24 = sext i32 %28 to i64
%memref-basereg25 = add i64 %memload22, %RCX24
%29 = inttoptr i64 %memref-basereg25 to ptr
%memload26 = load i32, ptr %29, align 1
%30 = trunc i32 %memload26 to i8
%EDI = sext i8 %30 to i32
%ESI27 = mul nsw i32 %ESI21, %EDI
%memload28 = load i64, ptr %RBP_N.32, align 1
%memload29 = load i64, ptr %stktop_4, align 1
%31 = trunc i64 %memload29 to i32
%RCX30 = sext i32 %31 to i64
%memref-basereg31 = add i64 %memload28, %RCX30
%32 = trunc i32 %ESI27 to i8
%33 = inttoptr i64 %memref-basereg31 to ptr
store i8 %32, ptr %33, align 1
%memload32 = load i64, ptr %RBP_N.8, align 1
%memload33 = load i64, ptr %stktop_4, align 1
%34 = trunc i64 %memload33 to i32
%RCX34 = sext i32 %34 to i64
%memref-basereg35 = add i64 %memload32, %RCX34
%35 = inttoptr i64 %memref-basereg35 to ptr
%memload36 = load i32, ptr %35, align 1
%36 = trunc i32 %memload36 to i8
%EDI37 = sext i8 %36 to i32
%memload38 = load i64, ptr %RBP_N.16, align 1
%memload39 = load i64, ptr %stktop_4, align 1
%37 = trunc i64 %memload39 to i32
%RCX40 = sext i32 %37 to i64
%memref-basereg41 = add i64 %memload38, %RCX40
%38 = inttoptr i64 %memref-basereg41 to ptr
%memload42 = load i32, ptr %38, align 1
%39 = trunc i32 %memload42 to i8
%R8D = sext i8 %39 to i32
%EDI43 = sub i32 %EDI37, %R8D
%40 = call { i32, i1 } @llvm.usub.with.overflow.i32(i32 %EDI37, i32 %R8D)
%CF44 = extractvalue { i32, i1 } %40, 1
%ZF45 = icmp eq i32 %EDI43, 0
%highbit46 = and i32 -2147483648, %EDI43
%SF47 = icmp ne i32 %highbit46, 0
%41 = call { i32, i1 } @llvm.ssub.with.overflow.i32(i32 %EDI37, i32 %R8D)
%OF48 = extractvalue { i32, i1 } %41, 1
%42 = and i32 %EDI43, 255
%43 = call i32 @llvm.ctpop.i32(i32 %42)
%44 = and i32 %43, 1
%PF49 = icmp eq i32 %44, 0
%memload50 = load i64, ptr %RBP_N.40, align 1
%memload51 = load i64, ptr %stktop_4, align 1
%45 = trunc i64 %memload51 to i32
%RCX52 = sext i32 %45 to i64
%memref-basereg53 = add i64 %memload50, %RCX52
%46 = trunc i32 %EDI43 to i8
%47 = inttoptr i64 %memref-basereg53 to ptr
store i8 %46, ptr %47, align 1
%memload54 = load i64, ptr %RBP_N.8, align 1
%memload55 = load i64, ptr %stktop_4, align 1
%48 = trunc i64 %memload55 to i32
%RCX56 = sext i32 %48 to i64
%memref-basereg57 = add i64 %memload54, %RCX56
%49 = inttoptr i64 %memref-basereg57 to ptr
%memload58 = load i32, ptr %49, align 1
%50 = trunc i32 %memload58 to i8
%EAX = sext i8 %50 to i32
%memload59 = load i64, ptr %RBP_N.16, align 1
%memload60 = load i64, ptr %stktop_4, align 1
%51 = trunc i64 %memload60 to i32
%R9 = sext i32 %51 to i64
%memref-basereg61 = add i64 %memload59, %R9
%52 = inttoptr i64 %memref-basereg61 to ptr
%memload62 = load i32, ptr %52, align 1
%53 = trunc i32 %memload62 to i8
%R8D63 = sext i8 %53 to i32
%54 = sext i32 %EAX to i64
%55 = lshr i64 %54, 32
%EDX64 = trunc i64 %55 to i32
%56 = sext i32 %EAX to i64
%57 = sext i32 %EDX64 to i64
%div_hb_ls = shl nuw i64 %57, 32
%dividend = or i64 %div_hb_ls, %56
%58 = sext i32 %R8D63 to i64
%div_q = sdiv i64 %dividend, %58
%EAX65 = trunc i64 %div_q to i32
%div_r = srem i64 %dividend, %58
%EDX66 = trunc i64 %div_r to i32
%memload67 = load i64, ptr %RBP_N.48, align 1
%memload68 = load i64, ptr %stktop_4, align 1
%59 = trunc i64 %memload68 to i32
%R969 = sext i32 %59 to i64
%memref-basereg70 = add i64 %memload67, %R969
%60 = trunc i32 %EDX66 to i8
%61 = inttoptr i64 %memref-basereg70 to ptr
store i8 %60, ptr %61, align 1
%memload71 = load i32, ptr %stktop_4, align 1
%EAX78 = add i32 %memload71, 1
%62 = call { i32, i1 } @llvm.uadd.with.overflow.i32(i32 %memload71, i32 1)
%CF72 = extractvalue { i32, i1 } %62, 1
%63 = and i32 %EAX78, 255
%64 = call i32 @llvm.ctpop.i32(i32 %63)
%65 = and i32 %64, 1
%PF73 = icmp eq i32 %65, 0
%ZF74 = icmp eq i32 %EAX78, 0
%highbit75 = and i32 -2147483648, %EAX78
%SF76 = icmp ne i32 %highbit75, 0
%66 = call { i32, i1 } @llvm.sadd.with.overflow.i32(i32 %memload71, i32 1)
%OF77 = extractvalue { i32, i1 } %66, 1
store i32 %EAX78, ptr %stktop_4, align 1
br label %bb.1
```

```
bb.3:
ret void
```